# Extending the Kerberos Protocol for Distributed Data as a Service

Graham Bent
IBM UK
GrahamBent@uk.ibm.com

Dominic Harries
IBM UK
dharries@uk.ibm.com

Andrew J. Toth
US Army Research Laboratory
andrew.j.toth.civ@mail.mil

***Abstract*—Whilst much of the research on authentication in peer to peer networks focuses on distributed authentication services, in current military systems the use of a centralized authority, such as the Kerberos ticketing framework predominates. Kerberos v5 is targeted at giving users access to a specific service with the option of delegating credentials to other authenticated nodes to enable them to act as proxies to access the service. The model does not work in situations where there are many services, distributed across a rapidly changing network, which could respond to a single request. An example of such a distributed set of services is a Gaian Database, where the nodes represent distributed data services and the queries represent the service requests.**

**In this work we describe an extension to the Kerberos ticketing framework that provides the delegated credentials 'on demand' for nodes that can respond to the service request. We describe an implementation of the protocol that is used to enable authenticated policy-based access control using the Gaian Database to access distributed data sources in a military coalition scenario. The approach has been demonstrated in support of a Coalition Warfare Program (CWP) demonstration held at the NATO International Fusion Centre (IFC) at RAF Molesworth UK.**

## I. Introduction

An important aspect of the ITA research program is the transition of research to higher technology readiness levels. In 2011 the Office of the Secretary of Defense (OSD)'s Coalition Warfare Program (CWP) funded a 2 year transition project on "ITA Policy Controlled Information Query & Dissemination". The goal of this ongoing CWP project is to develop an extensible capability to perform distributed federated query and information dissemination across a coalition network of distributed disparate data/information sources with access control policies.

The CWP project is led by US Army Research Laboratory (ARL) and UK Defence Science Technology Laboratory (DSTL) with software development by IBM UK and IBM US. The CWP project exploits two key technology components developed within the ITA, namely the Gaian Database and integrated Access Policy Decision and Enforcement (APDE) mechanisms. The Gaian Database (GaianDB) is a Dynamic Distributed Federated Database (DDFD) that addresses a need to share information among coalition members by providing a means for policy-controlled access to data across a network of heterogeneous data sources [1], [2]. The GaianDB implements a SQL-compliant Store-Locally-Query-Anywhere (SLQA) approach providing software applications with global access to data from any node in the database network via standard SQL queries. Data access and dissemination is controlled via APDE security policies which are enforced at the database node level, reducing potential for unauthorized data access and waste of network bandwidth. The end goal of the CWP project is to demonstrate the GaianDB and APDE policy technology within an operational environment at the NATO Intelligence Fusion Centre (IFC) based at RAF Molesworth in the UK.

In the context of a service oriented architecture the Gaian Database provides an efficient layer for accessing and performing database operations on distributed data sources, the owners of the different data sources, which we term 'Data as a Service'. Access to the data is usually controlled by the owners of the different data sources who will have different policies for access to and the dissemination of the data that they hold. In rapidly changing military operations and particularly in coalition operations there is a need to negotiate and adjust these policies among coalition elements in tactical time scales. This requires capabilities for automatic refinement of high-level goals into enforceable system controls, and formal analysis to ensure that policies are correct and consistent, conflict-free, and enforceable with available mechanisms. The goal of policy based self-management is to enable development of software systems that can implement a subset of the reasoning processes of a human administrator in an automated manner.

The IBM Watson Policy Management Library (WPML) [3] is an example of a comprehensive framework for using policies in computer communication networks using a centralized model for policy checking and refinement. In previous work, WPML was adapted to enable distributed policy management for use with the Gaian Database [4]. The previous work showed that enforcing global data access policies across a DDFD can be achieved using a distributed version of WPML where the Policy Enforcement Point (PEP) and Policy Decision Point (PDP) components are deployed to all database nodes where policy is to be enforced. The concept is shown in Figure 1.

Each PDP has access to a distributed policy repository, which uses a local database connection for writes (Store Locally) and a Gaian database connection for reads (Query Anywhere). A Policy Management Tool can be used to create, check and store policies at any node in the network. These policies can then be read and enforced by the PEPs on all relevant nodes in the network. The number of nodes that are

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*



| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED |
|---|---|---|
| **20 SEP 2012** | | **00-00-2012 to 00-00-2012** |

| 4. TITLE AND SUBTITLE | 5a. CONTRACT NUMBER |
|---|---|
| **Extending the Kerberos Protocol for Distributed Data as a Service** | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **US Army Research Laboratory,2800 Powder Mill Road,Adelphi,MD,20783-1197** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) |
| | 11. SPONSOR/MONITOR'S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT
**Approved for public release; distribution unlimited**

13. SUPPLEMENTARY NOTES
**Presented at the Annual Conference of International Technology Alliance (ACITA) 2012 held 17-21 Sep in Southampton, UK. U.S. Government or Federal Rights License**

14. ABSTRACT
**Whilst much of the research on authentication in peer to peer networks focuses on distributed authentication services, in current military systems the use of a centralized authority such as the Kerberos ticketing framework predominates. Kerberos v5 is targeted at giving users access to a specific service with the option of delegating credentials to other authenticated nodes to enable them to act as proxies to access the service. The model does not work in situations where there are many services distributed across a rapidly changing network, which could respond to a single request. An example of such a distributed set of services is a Gaian Database, where the nodes represent distributed data services and the queries represent the service requests. In this work we describe an extension to the Kerberos ticketing framework that provides the delegated credentials ?on demand? for nodes that can respond to the service request. We describe an implementation of the protocol that is used to enable authenticated policy-based access control using the Gaian Database to access distributed data sources in a military coalition scenario. The approach has been demonstrated in support of a Coalition Warfare Program (CWP) demonstration held at the NATO International Fusion Centre (IFC) at RAF Molesworth UK.**

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT **unclassified** | b. ABSTRACT **unclassified** | c. THIS PAGE **unclassified** | **Same as Report (SAR)** | **7** | |

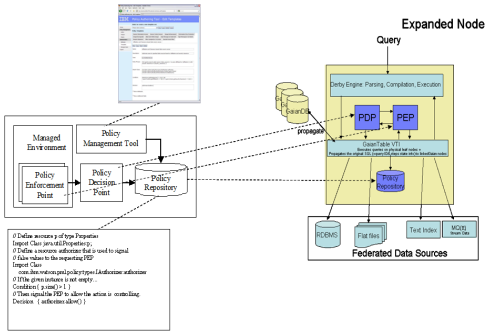


Fig. 1. Distributed Policy Management using WPML

chosen to enforce policies in the network represents a trade-off between robustness of policy enforcement and overhead of policy distribution: the larger the number of nodes that enforce policies the tighter the control of data dissemination. However, policies have to be transferred to a larger number of nodes, to be analyzed and enforced, which increases the overhead of transmitting them over the network using the Query Anywhere mechanism. Further details can be found in [4].

One of the essential elements in applying any policy is the capability to securely authenticate the originator of the service request so that the appropriate policies can be applied. Whilst much of the research on authentication in peer to peer networks focuses on distributed authentication services, in many military organizations such as the NATO IFC, the use of a centralized authority, such as the Kerberos ticketing framework predominates. Kerberos v5 is targeted at giving users access to a specific service with the option of delegating credentials to other authenticated nodes to enable them to act as proxies to access the service. However, the model does not work in situations where there are many services, distributed across a rapidly changing network, which could respond to a single request. The Gaian Database is an example of such a distributed set of services, where the Gaian nodes represent distributed data services and the queries represent the service requests. In this paper we describe an extension to the Kerberos ticketing framework that provides the delegated credentials 'on demand' for nodes that can respond to the service request. We then describe how the extended protocol is being implemented in a demonstration system at the NATO IFC.

## II. Related Work

Kerberos [5] is an authentication protocol for distributed networks of computers. Based on the Needham-Schroder Symmetric Key Protocol [6], it has gained widespread popularity and is supported on a wide range of platforms including Windows®, Linux®, Solaris®, AIX®, and z/OS®.

The protocol is based on the use of tickets, which are short-lived credentials passed to services that a user would like to access. These tickets are obtained from a central server, known as the Authentication Server (AS) and are used to authenticate the user to specified services which have also registered with the AS.

From its inception it was noted that traditional Kerberos presents a number of challenges in terms of its scalability and the potential vulnerability of a centralized certificate authority. This gave rise to significant research into alternative approaches that are better suited to a peer to peer authentication with various public key cryptographic (PKC) services being devised, such as identity based PKC (ID-PKC) [7] and more recently certificateless schemes (CS-PKC) [8]. Nevertheless Kerberos has stood the test of time and given its wide range of platform support, particularly Windows, it is not surprising to find that is widely adopted as the authentication mechanism in many military systems.

One of the challenges in introducing new technologies, such as the Gaian Database, into these existing military systems is that of obtaining the necessary accreditation. Introducing such technology requires an evolutionary rather than a revolutionary approach particularly in the area of security thus forcing the adoption of pragmatic approaches. In the case of the CWP program this required integration into a system that used the Kerberos authentication protocol. However the protocol is based on the assumption that the client requesting a service or services is aware of which services it needs to authenticate with. In the case of distributed data services this is generally not the case and an extension to the protocol is required. The following sections describe this extension and its implementation in a demonstration system at the IFC.

## III. Trust Management in Kerberos

The end result of the Kerberos protocol is the establishment of a shared key between a client and a service. The service can be sure that the client is genuine, and in standard deployments the client can be sure that the service is genuine. The shared key can subsequently be used to sign or encrypt messages between the client and the service.

We describe the interactions in the Kerberos protocol using the following notation:

| | |
|---|---|
| $K_A$ | $A$'s secret key |
| $K_{A,B}$ | Session key between $A$ and $B$ |
| $\{X\}_K$ | $X$ is encrypted with $K$ |
| $A \rightarrow B : M$ | Message $M$ sent from $A$ to $B$ |
| $ts_x$ | A timestamp |
| $addr$ | IP address of principal |
| $life$ | Lifetime of ticket (implemented as start time and end time) |

The messages described below have been simplified and do not include information such as the protocol version. For full specification of the messages, see RFC 1510 [9].

### A. Accessing a service using Kerberos

Before a client $C$ can access a Kerberos service $S$, it must receive a Ticket Granting Ticket (TGT) from the Authentication Service $AS$. The Authentication Service and the client have a shared secret $K_C$ — this could be a hash of the user's password. To begin authentication, the client sends its ID to Authentication Service:

$$C \rightarrow AS : C \tag{1}$$

The Authentication Service then responds with the TGT:

$$C \leftarrow AS : \underbrace{\{C, addr, life, K_{C,TGS}\}_{K_{TGS}}}_{T_{C,TGS}}, \{K_{C,TGS}\}_{K_C} \tag{2}$$

Now $C$ must decrypt the session key $K_{C,TGS}$ using its secret key $K_C$. $K_{C,TGS}$ is used to secure communication with the Ticket Granting Service (TGS). The TGS allows the client to request tickets for specific Kerberos services, and is on the same machine as the Authentication Service. $T_{C,TGS}$ is a Ticket Granting Ticket (TGT) encrypted with the private key of the TGS ($K_{TGS}$). The client $C$ cannot decrypt this, but during its lifetime (the default for Kerberos is 5 minutes) it allows the client to access services without re-authenticating (e.g. without re-entering a password).

To access a services, the client sends a message to the TGS which responds with a Service Ticket $T_{C,S}$:

$$C \rightarrow TGS : T_{C,TGS}, S, \{C, ts_1\}_{K_{C,TGS}} \tag{3}$$

$$C \leftarrow TGS : \underbrace{\{C, addr, life, K_{C,S}\}_{K_S}}_{T_{C,S}}, \{K_{C,S}\}_{K_{C,TGS}} \tag{4}$$

The user may then pass $T_{C,S}$ to the service $S$ for authentication:

$$C \rightarrow S : T_{C,S}, \{C, ts_2\}_{K_{C,S}} \tag{5}$$

$$C \leftarrow S : \{ts_2 + 1\}_{K_{C,S}} \tag{6}$$

Mutual authentication of $C$ and $S$ is obtained at this point.

### *B. Credential delegation*

Kerberos version 5 allows a user to delegate his/her credentials to an authenticated service. The service can then access other services as if it were the user. In order for a service to request a ticket for another service, it must be able to communicate with the TGS, i.e. it must know the session key between the client and TGS $K_{C,TGS}$. This can be obtained using the follow message, described in RFC 1510, section 5.8 [9]:

$$C \rightarrow S : T_{C,TGS}, \{K_{C,TGS}\}_{K_{C,S}} \tag{7}$$

Now $S$ can use the Ticket Granting Ticket $T_{C,TGT}$ to request tickets as described in messages 3 and 4 above. Normally the TGS would reject a request from an address which does not match that within the ticket granting ticket, but if the FORWARDABLE flag is set, it can ignore this discrepancy.

## IV. Extending Kerberos

Kerberos is designed to work in systems where a service that can fulfil a user's request is known in advance, and the set of services involved in one request is small and constant. When employed in a dynamic network where multiple unknown services could respond to one request, the design does not work so well. A ticket must be requested for every service that could be involved in a request, and authentication must be carried out with each service prior to (or during) the request. In an application like the Gaian Database, this set of services could be very large.

Our extension to Kerberos establishes a shared key between the client and the set of services that could respond to a specific request. Client identity assurance is provided at either the edge of a trusted network or throughout.

When dealing with an unknown set of services, it may not be the case that all services are trustworthy. If the request contains sensitive information, it should not be shared with any untrusted services. However, a service cannot determine whether it can respond to a request if the request is hidden. For this reason, we split the request into two parts, $R_{pub}$ which all nodes can see, and $R_{priv}$ which only authenticated nodes can see.

The Gaian Database provides a good example of a request that can be split in this way: an SQL query. Consider the query:

```
SELECT * FROM PEOPLE WHERE NAME = 'abdul'
```

This could easily be split into $R_{pub}$:

```
SELECT * FROM PEOPLE
```

Upon receiving this query, only Gaian nodes that had a `PEOPLE` table would need access to $R_{priv}$:

```
WHERE NAME = 'abdul'
```

Our extension dynamically authenticates services as a request is being processed. We assume that a user has authenticated initially with a single edge service $S$ using the standard Kerberos protocol (messages 1 – 6). A request is issued from this service to a dynamic set of services $D_{1...n}$:

$$S \rightarrow D_{1...n} : \{C\}_{K_Q}, \{R_{priv}\}_{K_Q}, R_{pub} \tag{8}$$

This message makes use of a query key $K_Q$ that is generated by the issuer of the request. It is recommended that this be generated by applying a one-way function to $K_{C,TGS}$ — this means that each user will get its own query key, and the expiry of the query key and the TGT will be linked. The same query key may be used for multiple requests, and services responding to a request can cache the key.

$R_{priv}$ has been encrypted with the query key, so initially none of the services will be able to decrypt it. A subset of the dynamic services, $D'$, will be able to ascertain that they could respond to the request based on information in $R_{pub}$:

$$D' = \{x \in D : can\_respond(D_x, R_{pub})\} \tag{9}$$

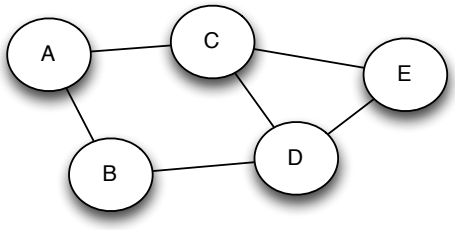


Fig. 2. Example network of nodes

A subset of these services, $D''$, will not have $K_Q$ cached from a previous request and will need to obtain it from the issuer of the request:

$$D'' = \{x \in D' : \neg cached(D_x, K_Q)\} \tag{10}$$

$$\forall x \in D''.(D_x \rightarrow S : D_x, \{C\}_{K_Q}) \tag{11}$$

The issuer of the request can then go through the standard steps of requesting a ticket for $D_x$ (messages 3 and 4) and then authenticating with $D_x$ (messages 5 and 6). During this process, the services will mutually authenticate each other, ensuring that the query key is protected. Then the issuer can send the query key query key to $D_x$, encrypted using the session key obtained with the ticket:

$$D_x \leftarrow S : \{K_Q\}_{K_{S,D_x}} \tag{12}$$

The node $D_x$ will then be able to decrypt $R_{priv}$ and ascertain whether it can respond to the request. If the response to a request is also sensitive, this may be encrypted with $K_Q$.

This scheme works when there is no private part of the query, here the propagation of $K_Q$ establishes a trusted path back to the issuer. It also works when there is no public part of the query, in this case all nodes will request $K_Q$.

### A. Efficient query key propagation in a network of nodes

The scheme described above works well when all the services that could respond to a request are connected directly to the service issuing the request. In dynamic networks, there may be one or more intermediate nodes through which the request passes, as shown in figure 2.

Let us consider a request $R$ from node A, where all the nodes have data relevant to the query. Under the simple scheme described above, node A would be contacted by B, C, D and E to obtain $K_Q$. The number of nodes contacting A scales linearly with the number of nodes in the network that have data. This situation can be improved by allowing intermediate nodes to pass $K_Q$ downstream.

In order to reach a specific service, $R$ will need to have travelled through a sequence of nodes. For example, to get to node E, some possible paths for $R$ would be (A, C, E) or (A, B, D, E).

If E needs to obtain the query key, it can ask the prior node in the sequence to supply it. The prior node then has several options:

1) If it knows $K_Q$, it can pass it downstream using the mechanism described above.
2) If it does not know $K_Q$, and does not want to find it out (because it doesn't have any relevant data), it can forward the request for $K_Q$ upstream, and act merely as a conduit for the ensuing key transportation. This case also works where the node is not trusted, because $K_Q$ is encrypted during transportation this does not introduce a security risk.
3) If it does not know $K_Q$ and would like to obtain it, it can make its own request upstream to obtain the key. Once this has happened, it can then proceed as in option 1.

Following this behaviour, the request will eventually reach the issuing node, which will always know $K_Q$.

### B. Delegation of authority through the network

A service in the dynamic network may need to be provided with the credentials of the client in order to process a request (e.g. to access a back-end data source). To achieve this, the same scheme as above may be used, but in addition to passing $K_Q$ as in message 12, the data from message 7 should also be passed:

$$D_x \leftarrow S : T_{C,TGS}, \{K_Q, K_{C,TGS}\}_{K_{S,D_x}} \tag{13}$$

This works with either of the proposed key propagation schemes.

An additional benefits of approach is that downstream services are authenticating with the client directly, this prevents an upstream service from spoofing a different client. See the following section for more information on this.

### C. Analysis of Attacks

Below we list potential attacks against the Kerberos extension and ways to mitigate these.

*1) Untrusted node modifies the query in transit:* In the scheme described above, an untrusted node through which the query is routed could alter the query that is being executed. This attack vector can be removed by generating a Message Integrity Code (MIC) using the query key, and appending this to the end of the message. A trusted node can then use this to verify that the query has not been tampered with.

*2) Trusted node modifies the query in transit:* It is not possible to defend against this attack. This is similar to the case in standard Kerberos where a service that has had a user's credentials delegated to it misuses these. One of the reasons why a query key is generated rather than using $K_{C,TGS}$ is to limit the possible damage from this scenario.

*3) Untrusted node generates a query with a fake query key:* The defense against this attack is based on the inter-node authentication that is performed when obtaining the query key. When a service receives the request, it will attempt to establish a secure connection with the query issuer to obtain a session key to transport the query key. At this point, the TGS would not allow the connection to be established with the untrusted node.

*4) Trusted node spoofs a query from a different user:* A trusted node can generate a fake query key for an arbitrary user and then send out a request posing as that user. This attack is not possible in the standard Kerberos protocol. There are several possible ways to mitigate it:

- Turn on credential delegation. This means that a downstream node can require current credentials for a user to be delegated before acting as that user.
- Incorporate PKI. A request could be signed with the private key of a user, and then verified on the downstream node using the public key of that user.
- Use $K_{C,TGS}$ directly rather than $K_Q$, and verify this with the TGS.

*5) Node pretends that it could respond to a query when it can't:* The main reason for including a stage where a node decides whether it can respond to a query are *a*) to reduce network traffic and minimise the load on the TGS; and *b*) to honour the 'need to know' security best practice. Even if it pretends that it could respond to a query, an untrusted node will not be able to obtain $K_Q$, because establishment of the secure connection will be denied.

*6) Trusted node generates query using cached query key:* If a node is compromised while it has a query key cached, it can issue queries to other nodes that have the same key cached even if its status is changed to untrusted by the TGS. This would only be possible for the lifetime of the cache, and could be mitigated by forbidding the caching of query keys.

## V. Implementation at the IFC

Like any other information consuming and producing organization, the NATO Intelligence Fusion Centre of Molesworth UK (IFC), is burdened with a set of inherent problems. The most notable being the breadth and depth of information available and capacity to process it in a timely manner. Identifying and addressing valuable information while ignoring less relevant information, all within a networked multinational coalition is a formidable task.

The issues involve key technology mainstays such as storage, access, security, search, retrieval and dissemination. Knowledge management remains a continuing challenge, especially in such an information rich environment driven by our modern methods of collection and storage. The continued reliance on "single use" search and lack of true federated data, mean most data users must access and collate multiple data sets during research. True subject matter experts are sometimes few and far between, the rate and pace of information requests often reduce mentoring capacity.

A single silver bullet to fix all key issues is unlikely at best, since each facet has its own discreet set of challenges. There are a multitude of software and storage solutions currently available but unless a significant leap forward is made organizations must consider there long term investments and interoperability before change is possible. There needs to be a way to share information better using existing infrastructure which empowers discovery, makes better use of network resources, and promotes security to include access and query logs to protect this improved data access.

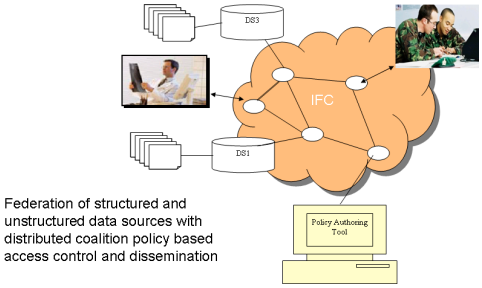


Fig. 3. Example NATO deployment of the Gaian Database to federate a wide range of coalition assets

The project as it relates to the IFC is primarily concerned with data discovery, the federation of various data sources with a major emphasis on the safeguarding of data access and dissemination. The approach that is being investigated is the use of distributed access policies and distributed query over multiple database enabled access nodes. Specifically, the work is considering the requirements and operation limitations which may be present when working in a NATO plus information sharing environment such as during Operation Unified Protector in Libya or ISAF in Afghanistan, this issue of authenticated federated sharing is a prominent and ongoing challenge.

The federation of the different data sources using the Gaian Database (figure 3) enables a range of new search and analysis applications to be exercised from anywhere in the network without the need for replication and with no central repository of information (i.e. a virtual knowledgebase). Previous work has demonstrated the federation of various data sources, which provide a mixture of both structured and unstructured information. These sources have included CIDNE, CSD, Tigr in representative emulated Afghan Mission Network scenario. The major contribution of the current work is the inclusion of an authentication mechanism to support a single-sign-on capability.

In the case of the IFC, the bulk of the data is in the form of textual reports. To federate the data from these reports with structured data held in other data repositories has required the development of a comprehensive text analytics capability that enables relevant information to be extracted. Natural Language Processing (NLP) rules were developed using the IBM LanguageWare [10] tool, and exported as a UIMA [11] PEAR file for deployment to IBM Content Analytics (ICA). A UIMA PEAR file is a deployable text analytics "pipeline" (analogous to a web application packaged in a WAR file). ICA is a text analysis and search application that supports UIMA.

The key entities targeted by NLP rules were people, places and organizations; and also general expressions of dates, times, and measurements. Rules were also created to cover specific entities such as IEDs, weapons, vehicles, etc. Also targeted were verb phrases indicating notable actions in the reports. These were aggregated with nearby dates, times and locations

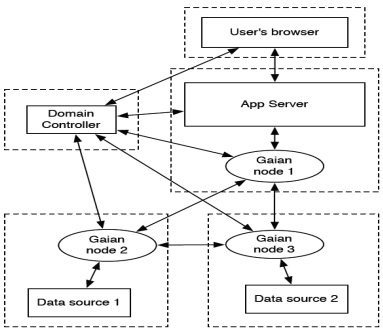


Fig. 4. IFC Security Architecture

to create "event" annotations.

The rule set developed is capable of linking entities to events, and by implication linking entities to each other "by event". In practice however, the bulk of the links created in the triple store are simple "mentioned in same sentence" links.

The extracted entity and link information is locally stored in a relational database triple store. The triple store and ICA text index are exposed through the local GaianDB node such that they can be queried to support a range of new and exisiting analytic applications (e.g. Federated Search & Query, Link Analysis) using tools such as the IBM i2 Analyst's Notebook.

### A. Security Considerations

The security infrastructure, as depicted in Figure 4, was based on the Kerberos network authentication protocol described in section III. Microsoft Windows ships with native support for this, and a Windows 2008 R2 Domain Controller was used as both an authentication server and a directory of users and their associated attributes. The primary means of interaction with the system was through a web interface. User authentication to this interface utilized the Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO) [12] which passed a Kerberos ticket to the web application server, rather than a username and password. The user only needed to authenticate once when logging on to the operating system, which would then negotiate with the Ticket Granting Service (Domain Controller) to obtain the ticket for the web application server.

Once the user had authenticated, the application server created the query key ($K_Q$) for that user. This key was generated by applying a one-way function to the user's Kerberos session key, and was then used to create message integrity codes for the queries performed by that user. In order to securely propagate the query key into the Gaian database, the Generic Security Services Application Program Interface (GSSAPI) [13] was used to establish a secure connection with the first node. The query key was then passed encrypted down this link. If a downstream Gaian node in the network (e.g. one connected to a data source) received a query that had been processed with a key that it didn't know, it requested a secure session with the originating Gaian node, which would then establish the session and securely transport the query key down to the node.

Once a node could verify the author of a query, it could utilise this information in the policies that were applied to data retrieved from that node. As well as the user ID, user affiliation and rank could also be considered in the policies. This information was obtained through an LDAP query of the Active Directory running on the domain controller, which had previously had these custom attributes added to the schema and populated for the users on the system.

## VI. Conclusion and Future Work

Many military organisations today have built their security infrastructure around the Kerberos authentication system. Although standard Kerberos does not enable secure authentication in a truely distributed system, such as the Gaian Database, replacing this infrastruture with peer to peer approaches is not a pragmatic solution in current operational environments.

In order to meet the objectives of the CWP transition programme, we have focused our research on extending the Kerberos protocol to operate within a dynamic network of services. By utilising 'on demand' propagation of a query key, we demonstrate how services in the network can ascertain whether they can respond to a request, while maintaining the security of a private part of that request. We propose a novel scheme for optimising the distribution of the query key.

The next phase of the CWP programme will focus on providing an operational capability to the IFC. To assist the accreditation process, a formal proof of security is advantageous. Burrows-Abadi-Needham (BAN) logic provides a mathematical framework within with such proofs can be developed. In the original paper [14], a proof of the basic authentication within the Kerberos protocol is provided. We intend to extend this proof to cover the extensions proposed in this paper.

## Acknowledgments

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

We acknowledge the support of Mr Todd Karr of the Knowledge Management Group at the IFC, who was the sponsor for this work.

## References

[1] G. Bent, P. Dantressangle, D. Vyvyan, A. Mowshowitz, and V. Mitsou, "A dynamic distributed federated database," in *Proc. 2nd Ann. Conf. International Technology Alliance*, 2008.

[2] G. Bent, P. Dantressangle, P. Stone, D. Vyvyan, and A. Mowshowitz, "Experimental evaluation of the performance and scalability of a dynamic distributed federated database," in *Proc. 3rd Ann. Conf. International Technology Alliance*, 2009.

[3] M. Beigi, S. Calo, and D. Verma, "Policy transformation techniques in policy-based systems management," in *Policies for Distributed Systems and Networks, 2004. POLICY 2004. Proceedings. Fifth IEEE International Workshop on*. IEEE, 2004, pp. 13–22.

[4] S. Calo, D. Wood, P. Zerfos, D. Vyvyan, P. Dantressangle, and G. Bent, "Technologies for federation and interoperation of coalition networks," in *Information Fusion, 2009. FUSION'09. 12th International Conference on*. IEEE, 2009, pp. 1385–1392.

[5] B. Neuman and T. Ts'o, "Kerberos: an authentication service for computer networks," *Communications Magazine, IEEE*, vol. 32, no. 9, pp. 33 –38, sept. 1994.

[6] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," *Commun. ACM*, vol. 21, no. 12, pp. 993–999, Dec. 1978. [Online]. Available: http://doi.acm.org/10.1145/359657.359659

[7] D. Boneh and X. Boyen, "Efficient selective-id secure identity-based encryption without random oracles," in *Advances in Cryptology-EUROCRYPT 2004*. Springer, 2004, pp. 223–238.

[8] S. Al-Riyami and K. Paterson, "Certificateless public key cryptography," *Advances in Cryptology-ASIACRYPT 2003*, pp. 452–473, 2003.

[9] J. Kohl and C. Neuman, "The Kerberos Network Authentication Service (V5)," RFC 1510 (Proposed Standard), Internet Engineering Task Force, Sep. 1993, obsoleted by RFC 4120. [Online]. Available: http://www.ietf.org/rfc/rfc1510.txt

[10] IBM languageware resource workbench. [Online]. Available: https://www.ibm.com/developerworks/community/alphaworks/lrw/

[11] Apache UIMA. [Online]. Available: http://uima.apache.org/

[12] L. Zhu, P. Leach, K. Jaganathan, and W. Ingersoll, "The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism," RFC 4178 (Proposed Standard), Internet Engineering Task Force, Oct. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4178.txt

[13] J. Linn, "Generic Security Service Application Program Interface Version 2, Update 1," RFC 2743 (Proposed Standard), Internet Engineering Task Force, Jan. 2000, updated by RFC 5554. [Online]. Available: http://www.ietf.org/rfc/rfc2743.txt

[14] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 426, no. 1871, pp. 233–271, 1989.